An extended Hamiltonian QR algorithm

Micol Ferranti *Bruno Iannazzo* *Thomas Mach*
Raf Vandebril

Report TW 667, May 2016

KU Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

An extended Hamiltonian QR algorithm

Micol Ferranti *Bruno Iannazzo* *Thomas Mach*
Raf Vandebril

Report TW 667, May 2016

Department of Computer Science, KU Leuven

Abstract

An extended QR algorithm specifically tailored for Hamiltonian matrices is presented. The algorithm generalizes the customary Hamiltonian QR algorithm with additional freedom in choosing between various possible extended Hamiltonian Hessenberg forms. We introduced in [Ferranti et al., *An extended Hessenberg form for Hamiltonian matrices*, Calcolo, available as TW665] an algorithm to transform certain Hamiltonian matrices to such forms. Whereas the convergence of the classical algorithm is related to classical Krylov subspaces, convergence in the extended case is related to extended Krylov subspaces, resulting in a greater flexibility, and possible enhanced convergence behavior. Details on the implementation, covering the bidirectional chasing and the bulge exchange based on rotations are presented. The numerical experiments reveal differences between the various extended forms and prove the validity of the approach.

Keywords : QR algorithm, Hamiltonian eigenvalue problems, extended Hessenberg matrices
MSC : Primary : 65F15 Secondary : 15A18, 15A23, 93B60.

AN EXTENDED HAMILTONIAN QR ALGORITHM*

MICOL FERRANTI‡, BRUNO IANNAZZO§, THOMAS MACH‡, AND RAF VANDEBRIL‡

Abstract. An extended QR algorithm specifically tailored for Hamiltonian matrices is presented. The algorithm generalizes the customary Hamiltonian QR algorithm with additional freedom in choosing between various possible extended Hamiltonian Hessenberg forms. We introduced in [Ferranti et al., *An extended Hessenberg form for Hamiltonian matrices*, Calcolo] an algorithm to transform certain Hamiltonian matrices to such forms. Whereas the convergence of the classical algorithm is related to classical Krylov subspaces, convergence in the extended case is related to extended Krylov subspaces, resulting in a greater flexibility, and possible enhanced convergence behavior. Details on the implementation, covering the bidirectional chasing and the bulge exchange based on rotations are presented. The numerical experiments reveal differences between the various extended forms and prove the validity of the approach.

Key words. QR algorithm, Hamiltonian eigenvalue problems, extended Hessenberg matrices

AMS subject classifications. 65F15, 15A18, 15A23, 93B60

1. Introduction.

A *Hamiltonian matrix* $\widehat{H} \in \mathbb{C}^{2n\times 2n}$ is a 2×2 block matrix defined as follows

$$\widehat{H} = \begin{bmatrix} \widehat{A} & \widehat{G} \\ \widehat{F} & -\widehat{A}^H \end{bmatrix}, \tag{1.1}$$

with $\widehat{F} = \widehat{F}^H$, $\widehat{G} = \widehat{G}^H$, and $\widehat{A}, \widehat{F}, \widehat{G} \in \mathbb{C}^{n\times n}$. The Hamiltonian structure has its impact on the spectrum, which is symmetric with respect to the imaginary axis. Classical dense eigenvalue solvers simply ignore this structure, but algorithms designed specifically for Hamiltonian matrices exploit this structure to gain in accuracy and speed [5, 12].

There are several approaches to this problem, the most direct of which is adapting the QR algorithm to the Hamiltonian structure. Unfortunately, designing such an algorithm is far from being trivial and so far, only the $\operatorname{rank}\widehat{F} = 1$ case has been worked out [12]. Another fruitful approach is based on forming a URV factorization of H [7, 14, 28, 36].

In this article we will focus on QR type algorithms. These algorithms have two main steps: there is a preprocessing step, in which the matrix is transformed to a convenient condensed form by unitary similarity transformation; and there is the actual processing step, in which the eigenvalues of the latter matrix are retrieved.

Using Francis's implicitly shifted QR algorithm (or QR algorithm for brevity) [17, 18, 38], the spectrum of $\widehat{H}$ can be computed in a backward stable manner. This

*The research was partially supported by the Research Council KU Leuven, projects CREA-13-012 Can Unconventional Eigenvalue Algorithms Supersede the State of the Art, OT/11/055 Spectral Properties of Perturbed Normal Matrices and their Applications, and CoE EF/05/006 Optimization in Engineering (OPTEC); by the Fund for Scientific Research–Flanders (Belgium) project G034212N Reestablishing Smoothness for Matrix Manifold Optimization via Resolution of Singularities; and by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization). The research of the second author was partly supported by INdAM through the GNCS Project 2016.

‡Department of Computer Science, KU Leuven, Celestijnenlaan 200A, 3001 Leuven (Heverlee), Belgium; ({micol.ferranti,thomas.mach,raf.vandebril}@cs.kuleuven.be).

§Dipartimento di Matematica e Informatica, Università degli Studi di Perugia, Via Vanvitelli 1, 06123 Perugia, Italy; (bruno.iannazzo@dmi.unipg.it).

means that the computed eigenvalues will be the exact eigenvalues of a nearby matrix $\widetilde{H}$, with $\widetilde{H}$ not necessarily of Hamiltonian structure. Hence the symmetry with respect to the imaginary axis is not necessarily preserved. Byers's Hamiltonian QR algorithm [12] is a structured variant of the QR algorithm preserving the Hamiltonian structure at each step of the algorithm and thus the symmetry of the spectrum. This results in a strongly backward stable algorithm [29]: the computed eigenvalues will be the exact eigenvalues of a nearby Hamiltonian matrix. A strongly backward stable algorithm usually yields a better accuracy, since the structured condition number of the Hamiltonian eigenvalue problem can be much smaller than the unstructured condition number [5]. Moreover, the Hamiltonian QR algorithm roughly halves the required storage and the number of required floating point operations [12]. In the real case a double shift version of the Hamiltonian QR algorithm preserving also the spectral symmetry with respect to the real axis can be designed [12].

Hamiltonian matrices are related to the numerical solution of algebraic Riccati equations and can be used in several applications, the most noticeable of which are in control [23, 27]. For further applications and references we refer the reader to the recent book by Bini, Iannazzo, and Meini [8].

This paper aims at generalizing the Hamiltonian QR algorithm to an extended Hamiltonian QR algorithm. Extended QR algorithms are a recent generalization of QR algorithms. While the classical QR algorithm links to Krylov subspaces [38], the extended QR algorithm connects to extended Krylov subspaces [30, 31]. Comparing the QR algorithm with the extended QR algorithm, there are two important differences [30, 31]:

- Replace the Hessenberg matrix by its QR factorization, QR, with R upper triangular and $Q = Q_1 \cdots Q_{n-1}$, where Q_i is a (Givens) rotation acting on the rows i and $i+1$. The bulge, which was an additional nonzero entry in the lower triangular part, becomes an additional rotation. The additional rotation does not fit the original pattern and is called the *misfit*.
- Replace the Hessenberg matrix A by an extended Hessenberg matrix. Thus Q can still be factorized into $n-1$ rotations, but these rotations are ordered arbitrarily, i.e., $Q = Q_{\sigma(1)} \cdots Q_{\sigma(n-1)}$, with σ a permutation of $(1, \ldots, n-1)$.

A more detailed introduction to extended QR algorithms is given in Subsection 1.4.

Also the extended Hamiltonian QR algorithm is a two step process. First, one brings the matrix into a condensed form, e.g., into upper (Hamiltonian) Hessenberg form. The reduction of a Hamiltonian matrix with rank $\widehat{F} = 1$ to extended Hamiltonian Hessenberg form has been described recently in [15]. Secondly, QR iterations preserving the condensed form are performed: this step will be described in this paper. Note, that there are certain applications in which extended Hamiltonian Hessenberg matrices arise naturally [16]. In this case the first step can be skipped.

The paper is organized as follows. In the remainder of this section we will briefly review the (K)-Hamiltonian structure, unitary core transformations, and the extended QR algorithm. In Section 2 we will present the extended (K)-Hamiltonian QR iteration. This will be followed by a short section on implementation details. In Section 4 we present some numerical experiments. The paper is concluded with Section 5.

In the rest of the paper we use the following notation: We denote the identity

matrix of size n by I_n and the flip matrix of size n by

$$\Phi_n = \begin{bmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{bmatrix}.$$

We will write I and Φ, when the size is clear from the context, and we will denote by e_j the j-th column of the identity matrix. We recall that a matrix is said to be J- or *per-Hermitian* if ΦA is Hermitian [11].

The matrix M^H is the Hermitian conjugate of M, while with $M(i:j, k:\ell)$ we address the submatrix with row indices $\{i, i+1, \ldots, j\}$ and column indices $\{k, k+1, \ldots, \ell\}$ following MATLAB notation.

1.1. K-Hamiltonian structure. To ease the description of the algorithm we will use K-Hamiltonian matrices, instead of Hamiltonian ones. A matrix $H \in \mathbb{C}^{2n\times 2n}$ is said to be *K-Hamiltonian* if $\widehat{H} = KHK$ is a Hamiltonian matrix, with

$$K = \begin{bmatrix} I_n & 0 \\ 0 & \Phi_n \end{bmatrix}. \tag{1.2}$$

The K-Hamiltonian structure is a simple permutation of the Hamiltonian structure, which allows us to simplify the algorithm and link it back in an easy manner to the QR and extended QR algorithm [31, 37]. The definition of K-Hamiltonian matrices leads to the following corollary.

COROLLARY 1.1. *Let $H \in \mathbb{C}^{2n\times 2n}$ be a K-Hamiltonian matrix. Then H admits the following block structure*

$$H = \begin{bmatrix} A & G \\ F & -\Phi A^H \Phi \end{bmatrix},$$

with $G\Phi$ and ΦF Hermitian and $A, F, G \in \mathbb{C}^{n\times n}$.

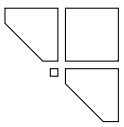
Proof. From the definition we have that $\widehat{H} = KHK$ is a Hamiltonian matrix, and thus

$$\widehat{H} = KHK = \begin{bmatrix} I & 0 \\ 0 & \Phi \end{bmatrix} \begin{bmatrix} A & G \\ F & -\Phi A^H \Phi \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & \Phi \end{bmatrix} = \begin{bmatrix} A & G\Phi \\ \Phi F & -A^H \end{bmatrix}. \qquad \square$$

It follows that F and G are per-Hermitian, i.e., $G\Phi = (G\Phi)^H = \Phi G^H$.

A K-Hamiltonian matrix H with A of upper Hessenberg form and $F = \alpha e_1 e_n^T$ is named a *K-Hamiltonian upper Hessenberg* matrix, which pictorially looks like

$$H = \begin{bmatrix} A & G \\ F & -\Phi A^H \Phi \end{bmatrix} = \quad .$$



A practical advantage of the K-Hamiltonian structure, over the Hamiltonian one, is that every K-Hamiltonian upper Hessenberg matrix is also of upper Hessenberg form.

Following the definition of the K-Hamiltonian matrix above, we call the matrix $S \in \mathbb{C}^{2n\times 2n}$ *K-symplectic* if KSK is symplectic, where $\widehat{S} \in \mathbb{C}^{2n\times 2n}$ is *symplectic* if $\widehat{S}^H \left[\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}\right] \widehat{S} = \left[\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}\right]$. Every K-Hamiltonian matrix with $\operatorname{rank} F = 1$ can be transformed into a K-Hamiltonian upper Hessenberg matrix by unitary K-symplectic similarity transformations, as shown in [1].

We can give a simple characterization of unitary K-symplectic matrices which will be useful in the following.

THEOREM 1.2. *A unitary matrix $S \in \mathbb{C}^{2n\times 2n}$ is K-symplectic if and only if it can be written as*

$$S = \begin{bmatrix} U_1 & U_2\Phi \\ -\Phi U_2 & \Phi U_1 \Phi \end{bmatrix}$$

for unitary matrices $U_1, U_2 \in \mathbb{C}^{n\times n}$. In particular, if S has the block diagonal structure

$$S = \begin{bmatrix} I_{n-1} & & \\ & G & \\ & & I_{n-1} \end{bmatrix},$$

where $G \in \mathbb{C}^{2\times 2}$, then

$$G = e^{i\psi} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

for some $\psi, \theta \in [0, 2\pi)$.

Proof. It is well-known that a unitary symplectic matrix T has the form $T = \left[\begin{smallmatrix} U_1 & U_2 \\ -U_2 & U_1 \end{smallmatrix}\right]$ for $U_1, U_2 \in \mathbb{C}^{n\times n}$; see, for instance, [8, Thm. 1.15]. Observing that KSK is unitary and symplectic we get the result. ∎

As a consequence of Theorem 1.2, real rotations whose active part involves the rows n and $n+1$ are K-symplectic.

A special case of K-symplectic matrices are block diagonal K-symplectic matrices, which have the form

$$S = \begin{bmatrix} S_{11} & 0 \\ 0 & S_{22} \end{bmatrix}, \quad \text{with} \quad S_{22}^H = \Phi S_{11}^{-1} \Phi.$$

If, additionally, S is unitary, then $S_{22} = \Phi S_{11} \Phi$ and both, S_{11} and S_{22}, are unitary, in view of Theorem 1.2.

1.2. Unitary core transformation. In this section we define unitary core transformations and their K-symplectic generalizations. We will also briefly review the most important operations with these transformations.

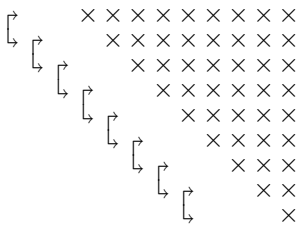
A *unitary core transformation* G_i is identical to the identity matrix except in the 2×2 submatrix $G_i(i : i+1, i : i+1)$, which is called the *active part*. Since G_i is unitary, the active part is unitary. In the remainder of this paper the index of a rotation is describing the position of the active part. A special subset of unitary core transformations are *rotations* with active part $\left[\begin{smallmatrix} c & -\overline{s} \\ s & \overline{c} \end{smallmatrix}\right]$, where $|c|^2 + |s|^2 = 1$. We will frequently depict a core transformation as $\overset{\curvearrowright}{\ }$, where the tiny arrows pinpoint the position of the active part. This will increase the readability as illustrated in the next example.

Example 1.3. The QR decomposition of an upper Hessenberg matrix $B \in \mathbb{C}^{n\times n}$ equals

$$B = QR = Q_1 Q_2 \cdots Q_{n-1} R,$$

where the matrices Q_i are rotations. Pictorially, by using the bracket notation, the QR decomposition is depicted as ($n = 9$)

$$B = QR = Q_1 Q_2 \cdots Q_{n-1} R =$$



.

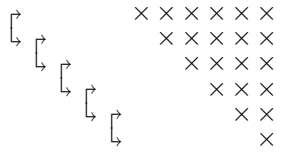
The matrix Q in Example 1.3 is written as a product of rotations, each of which acts on two consecutive rows. The order in which the rotations appears in the factorization of Q can be arbitrary.

Example 1.3 exhibits a descending *pattern* of rotations; with *pattern* we refer to the mutual positioning of the rotations which can be arbitrary and will be described by a position vector $\boldsymbol{p} \in \{\ell, r\}^{n-2}$ defined as follows:

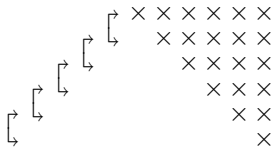
$$p_i = \begin{cases} \ell, & \text{if } Q_i \text{ is on the left of } Q_{i+1}, \\ r, & \text{if } Q_i \text{ is on the right of } Q_{i+1}. \end{cases}$$

Two factorizations in rotations share the same pattern if they can be ordered so that the graphical representation by brackets exhibits the same pattern. Note that this definition does not imply equality of the rotations but only their position in the factorization of Q. Note further that there are often multiple permutations σ with $Q = Q_{\sigma(1)} \cdots Q_{\sigma(n-1)}$, which lead to the same pattern.

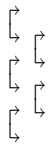
The rotations in the QR factorization of the upper Hessenberg matrix are ordered according to $p = (\ell, \ldots, \ell)$. An inverse Hessenberg matrix links to $p = (r, \ldots, r)$, the position vector of a unitary CMV matrix[1] equals $p = (\ell, r, \ell, r, \ldots)$. The following pictorial representations show these matrices and an arbitrary unstructured position vector.



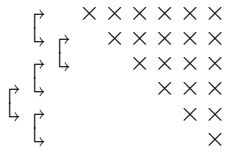
Hessenberg matrix



inverse Hessenberg matrix



unitary CMV matrix



$p = (\ell, r, r, \ell)$

As rotations acting on disjoint rows commute, there is no ambiguity in putting rotations on top or below each other as in the CMV or arbitrary case presented above.

All QR factorizations, with Q a product of $n-1$ rotations according to a certain position vector require an identical amount of storage. These matrices are called

[1] CMV matrices are already around for a long time, e.g., [21, 32], but they acquired their name from the initials of the authors of [13].

extended Hessenberg matrices and as proved in [30], all of these matrices are equally suited for a so-called *extended* QR algorithm, requiring $\mathcal{O}(n^2)$ storage and $\mathcal{O}(n^3)$ flops [30]. The name extended refers to the link with extended Krylov subspaces, see [24] for details.

1.3. Unitary symplectic core transformation. A unitary symplectic core transformation $G_i^{\widehat{S}}$ for $i < n$ can be defined as the product of two unitary core transformations $G_i G_{n+i}$, with equal active parts $G_i(i:i+1, i:i+1) = G_{n+i}(n+i : n+i+1, n+i : n+i+1)$. For the extended Hamiltonian QR algorithm, which we present in this paper by operating on K-Hamiltonian matrices, we need unitary K-symplectic core transformation $G_i^S = G_i G_{2n-i}$ for $i < n$, with active parts fulfilling $G_i(i:i+1, i:i+1) = \Phi G_{2n-i}(2n-i : 2n-i+1, 2n-i : 2n-i+1)\Phi$. When $i = n$, the active part of a K-symplectic unitary core transformation $G_n^S = G_n$ is located in the block $(n : n+1, n : n+1)$ and by Theorem 1.2 $G_n(n : n+1, n : n+1) = e^{i\psi} \left[\begin{smallmatrix} c & -s \\ s & c \end{smallmatrix}\right]$, with $\psi, c, s \in \mathbb{R}$. In particular $G_n(n : n+1, n : n+1)$ could be a real rotation.

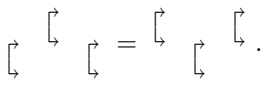
While a K-symplectic core transformation acts on rows n and $n+1$, a similar construction for symplectic matrices would act on rows n and $2n$. The latter would not fit our simple construction based on transformations acting on two consecutive rows. This and the fact that there exist irreducible K-Hamiltonian matrices in irreducible Hessenberg form are the main reasons why we use K-symplectic and K-Hamiltonian matrices.

Since K-symplectic rotations have a simple structure, we implement the algorithm with rotations, instead of generic core transformations, and restrict our investigations in the remainder of this paper to rotations.

In order to design the extended QR algorithm, we need some ways to manipulate them: we use the *fusion (to fuse)*, the *turnover (to turn over)*, and the *transfer through an upper triangular (to transfer through)* operations.

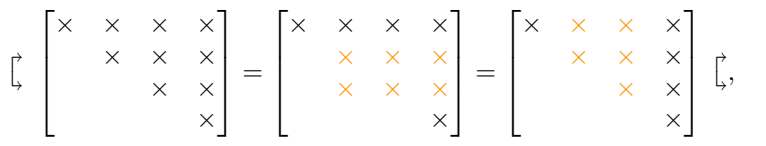
Two rotations acting on the same rows can be *fused*, $\curvearrowright\curvearrowright = \curvearrowright$. The result of a fusion of two rotations is again a rotation. As a result, also the product of two (K-)symplectic rotations acting on the same rows is a (K-)symplectic rotation.

Three rotations acting on three subsequent rows define a 3×3 unitary matrix. This unitary matrix can be factored in two different ways:



In fact, going from left to right we *turn over* the pattern of the rotations. Again, we can generalize this to (K-)symplectic rotations with $i \neq n$, where two turnovers, one in the lower and one in the upper half are executed simultaneously.

If we apply a rotation from the left to an upper triangular matrix, then an unwanted non-zero entry in the lower triangular part is created. This non-zero entry can be removed by pulling out a rotation from the right. Graphically this process can be depicted as



where the orange (light gray) entries are changed in the steps. This operation will be used in both directions; from left to right and from right to left and is *the transfer*

through an upper triangular operation. This operation extends naturally to (K-)symplectic rotations.

1.4. Extended QR algorithm. In this subsection the extended QR algorithm is presented briefly. For simplicity we restrict ourselves to explain the complex single shift case only. For further details we refer the reader to [30] and [31].

1.4.1. Chasing misfits instead of bulges. The first step towards an extended QR algorithm is to replace the Hessenberg matrix by its QR decomposition. We will explain the algorithm by operating simultaneously on the QR factorization $A = QR = Q_1 \cdots Q_{n-1} R$ and on the upper Hessenberg matrix A itself. We use the convention of Section 1, thus Q_i acts on rows i and $i+1$.

The QR iteration starts by picking a shift μ, typically an eigenvalue of the trailing 2×2 submatrix [39], and by computing a bulge or misfit generating rotation B_1 that fulfills

$$B_1^H \left(Q_1 \begin{bmatrix} r_{11} \\ 0 \\ \vdots \\ 0 \end{bmatrix} - \begin{bmatrix} \mu \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right) = B_1^H \begin{bmatrix} a_{11} - \mu \\ a_{21} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Next, the similarity transformation $B_1^H Q R B_1 = B_1^H A B_1$ is executed. Pictorially, for $n = 4$ we have

$$\begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}.$$

In the classical Hessenberg case the matrix multiplication is performed explicitly and a bulge, shown in the right-hand side of (1.3) is created. On the left side, we retain a factored form. The rotation B_1^H is therefore fused with Q_1, and we shift the rotation B_1 on the right, the misfit, through the upper triangular matrix. Pictorially, we end up with

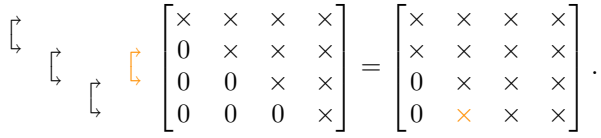
$$\begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}. \tag{1.3}$$

Next we will try to chase the bulge on the right-hand side and the misfit on the left-hand side. To do so, we first perform a turnover on the left and on the right we annihilate the bulge by pulling out a rotation.

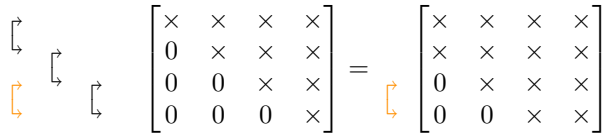
$$\begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}.$$

The leftmost rotations in both terms are essentially identical

and can be removed simultaneously by a single similarity transformation. New rotations emerge on the right of both matrices. Next, we shift the rotation through the upper triangular or apply it to the upper Hessenberg matrix. Pictorially, we have



An identical procedure as before, a turnover on the left and pulling out a rotation on the right, leads to



A last chase, a similarity transformation, a transfer through upper triangular, and a fusion, restore the upper Hessenberg form on the left. Also on the right after the similarity and multiplying out the factors we get a new Hessenberg matrix.

On the right hand side repeating this process leads to smaller and smaller subdiagonal elements until one or more become small enough to set them to zero and to deflate the problem into smaller subproblems. In the factored form deflations are signaled by almost diagonal rotations [26]. Proofs of convergence and more references can be found in [38] and [19]. Both algorithms are essentially the same, though the heuristics utilized typically differ; an analysis can be found in [26].

1.4.2. Extended QR iteration. The extended QR algorithm works essentially in the same way as the misfit chasing presented in Section 1.4.1. We execute an initial similarity transformation creating an auxiliary rotation, we chase the auxiliary rotation, and finally get rid of it by a fusion.

More in detail we get the following steps.

Misfit generation. Compute

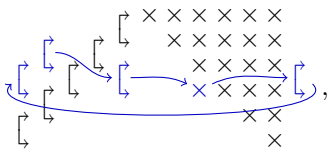
$$\begin{aligned} x &= (A-\mu I)e_1 = Q_1 \begin{bmatrix} r_{11} \\ 0 \end{bmatrix} - \begin{bmatrix} \mu \\ 0 \end{bmatrix}, && \text{if} \quad p_1 = \ell, \quad \text{or} \\ x &= (I-\mu A^{-1})e_1 = e_1 - \mu R^{-1} Q_1^H e_1 && \text{if} \quad p_1 = r. \end{aligned} \tag{1.4}$$

Thus in both cases only Q_1 and few entries of R are required. Once x has been obtained, then compute the rotation B_1^H with

$$B_1^H x = \begin{bmatrix} \times \\ 0 \end{bmatrix}. \tag{1.5}$$

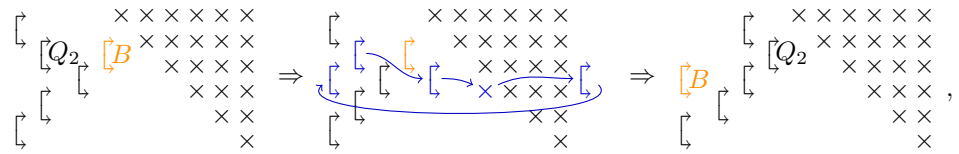
After applying the similarity transformation obtained by B_1 one of the two rotations can be fused and the other one is the misfit.

Misfit chasing. So far, we have only seen how misfit chasing on descending sequences works. The misfit chasing on ascending sequences is similar but the misfit is moved now from the left to the right. A step of the flow can be depicted as

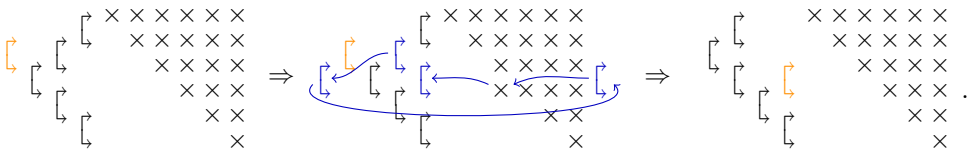


where the arrows describe the path the misfit follows.

It remains to describe what happens at a *bend*, i.e., a transition from ℓ to r or from r to ℓ in the position vector. A bend from descending (ℓ) to ascending (r) and the associated chasing step can be depicted as



where the role of the misfit B, originally the orange (light gray) rotation, is overtaken by Q_2, moving thereby the bend up one position. The bend from ascending to descending can be done analogously. Pictorially, we get:



We observe the following.

Remark 1.4. If a misfit is chased from top to bottom, then the pattern of rotations moves up one position. Here, the position of the last rotation Q_{n-1} can be chosen freely. A misfit can be chased, similarly, from bottom to top. In this case, the pattern of rotations moves down one position. Hence, the position of Q_1 can be chosen freely.

There is one important difference between the QR algorithm and the extended QR algorithm: whereas the convergence behavior for the QR algorithm is based on Krylov subspaces [38], the convergence behavior of the extended QR algorithm is based on extended Krylov subspaces [31]. Thus, the convergence behavior differs and a cleverly chosen combination of shifts and position vectors can accelerate convergence [30].

2. Extended (K-)Hamiltonian QR iteration. A Hamiltonian QR algorithm based on the QR algorithm is described in [12]. The main trick is to preserve the Hamiltonian structure by chasing two bulges, one with shift μ and one with shift $-\overline{\mu}$ at the same time. The Hamiltonian structure allows one to efficiently chase both bulges simultaneously. For simplicity, we restrict ourselves to the complex single shift case. In the K-Hamiltonian version one bulge starts at the top of the matrix and the other one starts at the bottom. Thus, the bulges meet in the middle where they have to be exchanged. In order to preserve the K-Hamiltonian structure, K-symplectic similarity transformations are used for the bulge generation, the chasing, and the bulge exchange.

In the case of an extended K-Hamiltonian QR algorithm the steps are analogous: we chase two misfits; the misfit generation depends on the first entry of the position vector. The chasing is similar to the procedure described in Section 1.4.2, however,

the misfit exchange in the middle requires some special care. According to Remark 1.4 the first misfit moves the bends up one row while the latter moves them down one row. Thus the pattern of rotations, except for the first one, stays the same.

In [15] we showed that an extended K-Hamiltonian Hessenberg matrix

$$H = \begin{bmatrix} A & G \\ F & -\Phi A^H \Phi \end{bmatrix}$$

is either of *descending* type

$$H = \begin{bmatrix} Q & 0 \\ 0 & I \end{bmatrix} G_n \begin{bmatrix} R & G \\ F & -\Phi R^H \Phi \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & \Phi Q^H \Phi \end{bmatrix}$$

or of *ascending* type

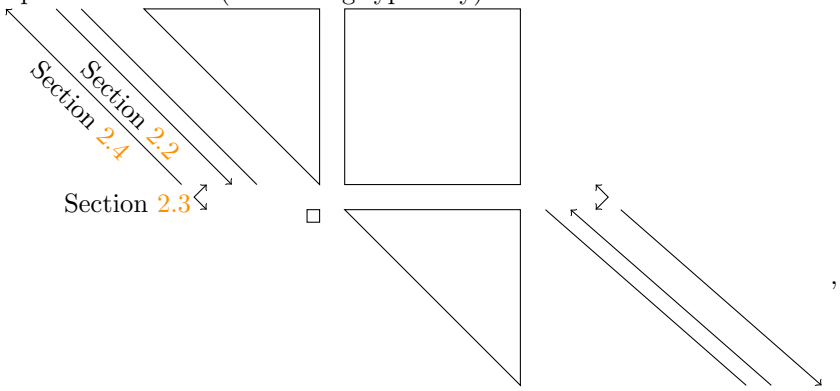
$$H = \begin{bmatrix} I & 0 \\ 0 & Q \end{bmatrix} G_n \begin{bmatrix} R & G \\ F & -\Phi R^H \Phi \end{bmatrix} \begin{bmatrix} \Phi Q^H \Phi & 0 \\ 0 & I \end{bmatrix},$$

with

$$Q = Q_{\sigma(1)} Q_{\sigma(2)} \cdots Q_{\sigma(n-1)} \qquad \text{and} \qquad F = f e_1 e_n^T.$$

Hence, an extended K-Hamiltonian Hessenberg matrix is completely described by $Q_1, \ldots, Q_{n-1}$, σ, $f = f_{1,n}$, the upper triangular matrix R, and the upper left triangular part of G, since G is per-Hermitian.

The algorithm described in the following sections have three main steps, which can be depicted as follows (descending type only)



where the sequences of rotations are represented by lines and the misfit chasing by arrows. The other cases can be handled similarly. In Section 2.1 we describe the creation of the misfits; in Section 2.2 we describe the chase of the misfits until they touch each other; in Section 2.3 we describe a way to swap the misfits; finally, in Section 2.4 we describe the chase of the misfits from the swap until their removal.

2.1. Misfit generation. First of all, we have to pick a shift. For the misfit chased from the top the Wilkinson shift is defined by the eigenvalues of $W = H(2n-1 : 2n, 2n-1 : 2n) = -\Phi A(1:2, 1:2)^H \Phi$. The Wilkinson shift μ is the eigenvalue of W that is closer to w_{22}. The misfit chased upward is linked to the shift $-\overline{\mu}$.

The misfit generation with the shift μ is identical to the misfit generation in the extended QR algorithm described in (1.4) and (1.5). We then apply the K-symplectic

similarity transformation defined by B_1. When the K-Hamiltonian Hessenberg matrix is of descending type we have

$$B_1^H \atop \Phi B_1^H \Phi \left(Q_{\sigma(1)} Q_{\sigma(2)} \cdots Q_{\sigma(n-1)} \begin{bmatrix} R & G \\ F & -\Phi R^H \Phi \end{bmatrix} \Phi Q_{\sigma(n-1)}^H \Phi \cdots \Phi Q_{\sigma(1)}^H \Phi \right) \begin{matrix} B_1 \\ \Phi B_1 \Phi \end{matrix}.$$

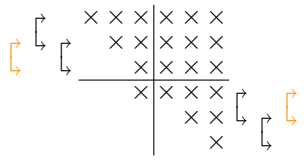
We transfer the misfit B_1 from right to the left. Since we perform everything only on the upper left part of the matrix, this implicitly shifts $\Phi B_1^H \Phi$ through the upper triangular from left to the right. Then we merge the two neighboring rotations in the first and last rows, as in the extended QR algorithm.

The case in which the K-Hamiltonian Hessenberg matrix is of ascending type can be treated similarly.

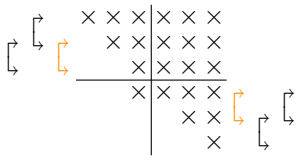
2.2. Misfit chasing (downward). The misfit is chased downward as in the extended QR algorithm, see Section 1.4.2. There is only one important point here: whenever we transfer a misfit through R we have to update G. Thereby we do not only have to update G with the rotation in the upper part but also with the linked rotation from the lower part of the matrix. By doing so the per-Hermitian structure of G is preserved.

2.3. Misfit exchange. We have chased the two misfits simultaneously to the middle of the matrix. Since the misfits are blocking each other further chasing is not possible. The misfit exchange will swap the shift information contained in the two misfits so that the chasing can be continued.

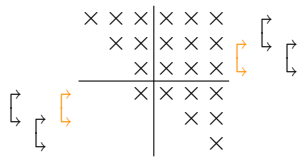
Depending on the last entry of the position vector and the chosen type, we are in one of the following four cases; the misfits B_{n-1} and $-\Phi B_{n-1}^H \Phi$ are marked orange (light gray):



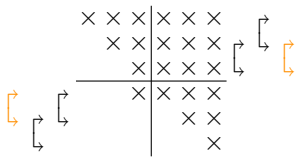
(1) ℓ, descending



(2) r, descending

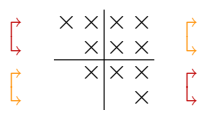


(3) ℓ, ascending



(4) r, ascending.

After a K-symplectic similarity transformation that brings the outermost two rotations to the other side (note that none of the rotations is blocked by other rotations), all four cases come down to the following:



$$= \left[\begin{array}{cc|cc} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ \hline x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{array}\right] = X, \tag{2.1}$$

where the two red (dark gray) rotations are linked and the two orange (light gray) ones, too. The matrix X is of K-Hamiltonian structure, and hence $x_{43} = -\overline{x}_{21}$ and $x_{33} = -\overline{x}_{22}$. Note also that the submatrix $X(3:4, 1:2)$ is of rank 1.

Let us assume for a moment that we are running a classical QR algorithm. We have chased a misfit downward to rows $n-1$ and n and have made a similarity transformation. The next chasing step would now involve a similarity transformation on rows n and $n+1$. However, we cannot perform this chasing step, since the misfit chased upward is blocking the other misfit. Looking at the misfit chased upward, we see that the next step is also a similarity transformation on rows n and $n+1$.

In the extended K-Hamiltonian QR algorithm we will apply a unitary K-symplectic similarity transformation to the middle rows of X, we use a real rotation matrix and hence form

$$Y = S_2^H X S_2 = \begin{bmatrix} 1 & & & \\ & c & s & \\ & -s & c & \\ & & & 1 \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} 1 & & & \\ & c & -s & \\ & s & c & \\ & & & 1 \end{bmatrix} \tag{2.2}$$

$$= \begin{bmatrix} x_{11} & cx_{12} + sx_{13} & \cdots \\ cx_{21} + sx_{31} & c^2x_{22} + csx_{32} + csx_{23} + s^2x_{33} & \cdots \\ cx_{31} - sx_{21} & c^2x_{32} - csx_{22} + csx_{33} - s^2x_{23} & \cdots \\ x_{41} & cx_{42} + sx_{43} & \cdots \end{bmatrix}, \tag{2.3}$$

with $c, s \in \mathbb{R}$. The result has to be of the same form as (2.1), and thus the submatrix

$$\begin{bmatrix} cx_{31} - sx_{21} & c^2x_{32} - scx_{22} + csx_{33} - s^2x_{23} \\ x_{41} & cx_{42} + sx_{43} \end{bmatrix} \tag{2.4}$$

has to be of rank 1. For $s = 0$ and $|c| = 1$ the rank is 1, but the shift information carried by the misfits would not be exchanged. The matrix F is of rank 1 if and only if

$$(cx_{31} - sx_{21})(cx_{42} + sx_{43}) = x_{41}(c^2x_{32} - scx_{22} + csx_{33} - s^2x_{23}).$$

The matrix $\left[\begin{smallmatrix} x_{31} & x_{32} \\ x_{41} & x_{42} \end{smallmatrix}\right]$ is of rank 1, hence $x_{41}x_{32} = x_{31}x_{42}$. Since $s \neq 0$ the equation above can be rewritten as

$$c(-x_{21}x_{42} + x_{31}x_{43} + x_{22}x_{41} - x_{33}x_{41}) + s(-x_{21}x_{43} + x_{23}x_{41}) = 0.$$

With $x_{43} = -\overline{x}_{21}$, $x_{33} = -\overline{x}_{22}$, and $x_{31} = \overline{x}_{42}$ we obtain

$$S_2 \begin{bmatrix} -x_{41}x_{23} - x_{21}\overline{x}_{21} \\ -x_{21}\overline{x}_{31} - \overline{x}_{21}x_{31} + (x_{22} + \overline{x}_{22})x_{41} \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}, \tag{2.5}$$

which defines S_2. Since $x_{41}, x_{23} \in \mathbb{R}$ the rotation S_2 is real and thus indeed a K-symplectic rotation, see Section 1.2.

For the misfit exchange one first has to form X; then one has to compute S_2 by (2.5), and the similarity transformation $M \to S_2^H M S_2$ has to be applied to the K-Hamiltonian matrix. Thereby updates involving columns of R and G are necessary. Finally, the updated matrix X can be factored again like in (2.1) retrieving the new misfit B_{n-1} and the updated rotation $Q_{\sigma(n-1)}$.

It remains to prove that the non-trivial rotation such that $\operatorname{rank} X(3:4,1:2) = \operatorname{rank} Y(3:4,1:2) = 1$ indeed swaps the shift information. Therefore we will first use the argument of Watkins [34] to retrieve the shift information stored in the bulge.

Let the matrix $\widetilde{A}$ be an intermediate matrix during a bulge chasing with block structure

$$\widetilde{A} = \begin{bmatrix} H_1 & * & * \\ & B_2 & * \\ & & H_3 \end{bmatrix}, \tag{2.6}$$

where $H_1 \in \mathbb{C}^{k+1,k}$ and $H_3 \in \mathbb{C}^{l+1,l}$ are of upper Hessenberg form or of upper Hessenberg possibly with some other bulge. Let N_{m+1} be a Jordan block of size $m+1$ for the eigenvalue 0. Watkins [34] showed that the eigenvalues of the pencil (B_2, N_{m+1}) are one infinite eigenvalue and the m shifts chosen for generating the bulge.

We are mainly interested in the single shift case, where $m = 1$. There the pencil (B_2, N_2) is of dimension 2×2, with $N_2 = \left[\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right]$. The matrix B_2 can be computed from the factored form by

$$B_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{smallmatrix} \nearrow \\ \searrow \end{smallmatrix} \begin{smallmatrix} \nearrow \\ \searrow \end{smallmatrix} R(k+1:k+3, k+1:k+2).$$

The leftmost rotation commutes with the projection matrix and hence we can apply a unitary congruence transformation to the pencil moving this rotation to N_2, we get the pencil

$$\left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{smallmatrix} \nearrow \\ \searrow \end{smallmatrix} R(k+1:k+3, k+1:k+2) \, , \begin{smallmatrix} \nearrow \\ \searrow \end{smallmatrix} \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \right),$$

where we can deflate the first row and column, which belong to the infinite eigenvalue. The remaining lower right entries of the pencil contain the actual shift information. Hence we can retrieve the shift by computing

$$\widehat{\mu} = -s_{Q_{k+2}} / s_B r_{k+2,k+2}, \tag{2.7}$$

where $s_{Q_{k+2}} = q_{k+2}(2,1)$ is the off-diagonal entry of the ith rotation in Q and s_B the off-diagonal entry of the misfit. One can easily show that (2.7) also holds if H_1 and H_3 are extended Hessenberg matrices. One can further show that in the inverse Hessenberg case the role of s_B and s_{Q_i} is interchanged in (2.7).

At the misfit exchange the two bulges are directly next to each other. Thus the above situation with $m = 1$ does not fully describe the situation. We thus need the following technical lemma, whose proof can be achieved by direct inspection.

LEMMA 2.1. *Let*

$$N = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \qquad A = \begin{bmatrix} a & b & g \\ f & e & -\overline{b} \\ h & \overline{f} & -\overline{a} \end{bmatrix} \in \mathbb{C}^{3\times 3},$$

with $g, e, h \in \mathbb{R}$, $h \neq 0$ *and* $|f|^2 - eh = 0$ *so that* $\operatorname{rank} \left[\begin{smallmatrix} f & e \\ h & \overline{f} \end{smallmatrix}\right] = 1$ *and let* $G = \left[\begin{smallmatrix} c & \overline{s} \\ -s & \overline{c} \end{smallmatrix}\right]$

be the rotation such that $G\begin{bmatrix} f \\ h \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$*, then* $s \neq 0$ *and*

$$\begin{bmatrix} 1 & 0 \\ 0 & G \end{bmatrix} (A - \lambda N) \begin{bmatrix} \Phi G^H \Phi & 0 \\ 0 & 1 \end{bmatrix} = R - \lambda \begin{bmatrix} -\overline{s} & \overline{c} & 0 \\ 0 & 0 & c \\ 0 & 0 & -s \end{bmatrix},$$

where R *is upper triangular and the eigenvalues of the pencil can be read from the diagonal as*

$$\lambda_u = -\frac{r_{11}}{\overline{s}} = \frac{bh - a\overline{f}}{h} = -\frac{1}{h} \det \begin{bmatrix} a & b \\ h & \overline{f} \end{bmatrix}, \qquad \lambda_c = \infty, \qquad \lambda_d = -\overline{\lambda}_u.$$

Applying Lemma 2.1 and the theory of Watkins to $X(2:4, 1:3)$ we get that $\lambda_u = -\overline{\mu} = -\frac{1}{x_{41}} \det(\left[\begin{smallmatrix} x_{21} & x_{22} \\ x_{41} & x_{42} \end{smallmatrix}\right])$ is the shift coming from the top, while $\lambda_d = \mu$ is chased upward. Indeed, the spectrum of $M(2:n, 1:n-1) - \lambda N_{n-1}$, where M is any of the matrices obtained during the chasing downward phase, is always $\{\infty, -\overline{\mu}\}$.

The same argument applied to $Y(2:4, 1:3)$ yields $\widetilde{\lambda}_u = -\frac{1}{y_{41}} \det(\left[\begin{smallmatrix} y_{21} & y_{22} \\ y_{41} & y_{42} \end{smallmatrix}\right])$, which is the information that will be chased upward by the next stages of the algorithm. In fact, the spectrum of $M(2:n, 1:n-1) - \lambda N_{n-1}$, where M is any of the matrices obtained during the chasing upward phase, is always $\{\infty, \widetilde{\lambda}_u\}$.

The swap has been done if $\widetilde{\lambda}_u = -\overline{\lambda}_u = \mu$. Using the explicit components of Y in (2.2), and with $c \neq 0$, we get that this fact follows from the choice of c and s in (2.5).

2.4. Misfit chasing (upward) and fusion. The misfit in the upper part of the matrix is now the misfit chased upward. This is equivalent to repeating the steps used to chase downward in reverse order.

At the end of the misfit chasing procedure, the misfit reaches the top of the matrix and we can fuse it into the pattern of rotations in two ways, allowing one to modify the first entry in the position vector of Q. In contrast with the one-directional chase in the extended QR algorithm, in the bi-directional chase the position vector stays the same except for the first entry which may change from one step to another.

2.5. Deflation. An extended K-Hamiltonian Hessenberg matrix is factorized as a product of rotations and a quasi-upper triangular K-Hamiltonian matrix

$$\begin{bmatrix} R & G \\ F & -\Phi R^H \Phi \end{bmatrix}$$

with R upper triangular and $F = fe_1 e_n^T$. In extended QR algorithms deflations are signaled by almost diagonal rotations [26]. Hence we will search for rotations with small subdiagonal entries. Additionally, there is the rare but very valuable deflation when $\| F \| = |f|$ becomes small. Hence a test of the form $\| F \| < \varepsilon(|h_{n,n}| + |h_{n+1,n+1}|)$ is performed. This criterion is tighter than $\| F \| < \varepsilon \| H \|_F$ and might lead to higher relative accuracy; see [22, Section 1.3.4]. In the K-Hamiltonian case this inequality simplifies with $\varepsilon(|h_{n,n}| + |h_{n+1,n+1}|) = 2\varepsilon |h_{n,n}|$ to $\| F \| < 2\varepsilon |h_{n,n}|$.

With each deflation the problem will deflate into three eigenvalue problems, $A(1:m, 1:m)$, $H(m+1:2n-m, m+1:2n-m)$, and $\left(-\Phi A^H \Phi\right)(n-m+1:n, n-m+1:n)$. Typically, m will be 1 or 2, and the eigenvalues of the first problem can be computed directly. In case m is larger the first and the last matrix are linked. If λ is an eigenvalue of $A(1:m, 1:m)$, then $-\overline{\lambda}$ is an eigenvalue of $\left(-\Phi A^H \Phi\right)(n-m+1:n, n-m+1:n)$. The eigenvalues of $A(1:m, 1:m)$ can be computed by an extended QR algorithm

[30, 31]. For $m \leq n-1$ the remaining K-Hamiltonian problem in the middle is again of extended K-Hamiltonian Hessenberg form which, for $m < n-1$, can be solved recursively by the same algorithm, while, for $m = n-1$, the eigenvalues can be computed by solving a quadratic equation.

3. Implementation details. We implemented the single shift extended Hamiltonian QR algorithm as described in the previous sections in MATLAB. Our MATLAB implementation is based on the well-tested and accurate core subroutines (generating rotation, fusion, and turnover) of the Fortran package **eiscor** [2]. Our implementation is available from .

Data storage. We represent the extended K-Hamiltonian Hessenberg matrix by its factored form, which is either (descending)

$$H = \begin{bmatrix} Q & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} R & G \\ F & -\Phi R^H \Phi \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & \Phi Q^H \Phi \end{bmatrix}$$

or (ascending)

$$H = \begin{bmatrix} I & 0 \\ 0 & Q \end{bmatrix} \begin{bmatrix} R & G \\ F & -\Phi R^H \Phi \end{bmatrix} \begin{bmatrix} \Phi Q^H \Phi & 0 \\ 0 & I \end{bmatrix},$$

with $Q = Q_{\sigma(1)} Q_{\sigma(2)} \cdots Q_{\sigma(n-1)}$, with σ according to position vector p. We store only f_{1n}, G, R, p, the type of the decomposition, ascending or descending, and the rotations in Q. For R and G we store the full square matrix. The code could be optimized by explicitly storing only the upper right triangular part of R and the upper left triangular part of G, since the other half is zero or defined by the per-Hermitian symmetry, respectively.

Rotations. For the implementation we use rotations $\left[\begin{smallmatrix} c & -s \\ s & \bar{c} \end{smallmatrix}\right]$ with real sine, $s \in \mathbb{R}$, and $|c|^2 + s^2 = 1$. For each rotation with real sine we only store three real values. Furthermore, rotations with real sines are advantageous, since a turnover of three rotations with real sines results again in three rotations with real sines. The result of a fusion of two rotations with real sines, however, is a rotation with a possible complex sine. By multiplying the rotation by a diagonal matrix the rotation can be transformed back into a rotation with real sine. The diagonal matrix can be passed through other rotations and merged into the upper triangular matrix R.

Observe that for rotations with a real sine, $s \in \mathbb{R}$, it holds that

$$\Phi Q_1^H \Phi = \Phi \begin{bmatrix} c & -s \\ s & \bar{c} \end{bmatrix}^H \Phi = \begin{bmatrix} c & -s \\ s & \bar{c} \end{bmatrix}.$$

Preparation steps. Additionally, one may choose to do two preparation steps. Eigenvalues should be deflated if the structure allows it, so that the resulting extended K-Hamiltonian Hessenberg matrix is unreduced. Further, the accuracy of the computations typically benefits from a balancing of the matrix that is a diagonal scaling, with some noticeable exception [35]. These steps have been investigated in [3] for dense matrices and in [4] for sparse matrices.

The examples chosen in the next section are unreduced and sufficiently balanced. Hence, we did not implement any additional deflation or balancing strategy.

4. Numerical experiments. We have tested our MATLAB implementation of the extended Hamiltonian QR algorithm on a compute server with two Intel Xeon E5645 CPUs running at 2.40 GHz with MATLAB version 8.5.0.197613 (R2015a). We

tested the accuracy of the bulge exchange in Section 4.1; the number of iterations per eigenvalue and the accuracy of the extended QR algorithm in Section 4.2; and the performance for different position vectors in Section 4.3. We further tested the code with two examples from the CAREX package [6] in Section 4.4.

4.1. Misfit exchange. In extended QR algorithms the shift information is encoded in the misfit. In floating point arithmetic this shift information is perturbed during the misfit chasing: the shifts get blurred. In some cases, e.g., in multishift implementations, the effect of shift blurring is so extreme that no useful shift information reaches the bottom of the matrix and the convergence stalls [33].

In the extended K-Hamiltonian QR algorithm we have an additional possible source of perturbations: the misfit exchange. In this numerical experiment we investigate the shift information stored in the misfit, at the beginning of the misfit chasing, directly before the misfit exchange, after the misfit exchange, and at the end of misfit chasing. To do so we extract the shift from the misfit as described in (2.7).

We generate three random matrices and compute the shift information stored in the misfit before and after the misfit exchange. We choose random extended K-Hamiltonian Hessenberg matrices of dimension 200×200 ($n = 100$). Therefore, we compute a random position vector, complex random matrices G and R, and real random rotations Q by the following MATLAB code:

```
G = rand(N,N) + 1i*rand(N,N);
G = G + G';
G = G(1:N,N:-1:1);
[unused,R]=qr(rand(N,N) + 1i*rand(N,N));
for i=1:N-1
  [Rot(1,i),Rot(2,i)]=givens(rand(1,1),rand(1,1));
end
```

The MATLAB function `givens` has been replaced by the more stable function `d_rot2_vec2gen(a,b)` from eiscor [2].

In Table 4.1 we can see that the misfit exchange is an additional source of perturbations but the effect is of the same order as the misfit chasing steps. If the matrix F is almost zero the misfit exchanging similarity transformation becomes almost diagonal. However, the misfits are still exchanged well. Also a small diagonal entry at the end of R does not affect the performance of the misfit exchange. Table 4.1 shows the absolute deviation of the shift from the intended one, after the generation of the misfit, after chasing the misfit one row down, directly before and after the misfit exchange, after chasing the misfit one row down after the misfit exchange, and before the misfit is fused at the end of the chasing procedure.

4.2. Iterations per eigenvalue and accuracy. In the last section we have seen that the shift information represented in the misfit reaches the bottom of the matrix without significant deterioration. Thus we expect that about four iterations per eigenvalue are necessary.

Table 4.2 shows the number of iterations divided by n for $n = 25, 50, 100$, and 200. For each n we generated 250 extended K-Hamiltonian Hessenberg matrices as described in Section 4.1; the resulting matrices are of size $2n \times 2n$. Since every iteration chases two misfits simultaneously, the number of iterations divided by n is comparable to the number of single shift iterations divided by the number of eigenvalues in the classical QR algorithm.

	random f_{1n}	$f_{1n} = 1\,\mathrm{e}{-10}$
after misfit generation	$9.1551\,\mathrm{e}{-16}$	$4.4458\,\mathrm{e}{-16}$
after one chase	$9.9301\,\mathrm{e}{-16}$	$3.3422\,\mathrm{e}{-16}$
before misfit exchange	$9.9301\,\mathrm{e}{-15}$	$8.3675\,\mathrm{e}{-15}$
after misfit exchange	$1.0187\,\mathrm{e}{-14}$	$8.5885\,\mathrm{e}{-15}$
after next chase	$8.2396\,\mathrm{e}{-15}$	$8.9207\,\mathrm{e}{-15}$
before fusion	$2.1284\,\mathrm{e}{-14}$	$1.6679\,\mathrm{e}{-14}$
	$f_{1n} = 1\,\mathrm{e}{-15}$	$r_{100,100} = 1\ \mathrm{e}{-10}$
after misfit generation	$8.8829\,\mathrm{e}{-16}$	$1.7764\,\mathrm{e}{-15}$
after one chase	$3.5531\,\mathrm{e}{-15}$	$3.5530\,\mathrm{e}{-15}$
before misfit exchange	$1.3361\,\mathrm{e}{-14}$	$3.5605\,\mathrm{e}{-15}$
after misfit exchange	$1.6028\,\mathrm{e}{-14}$	$8.8836\,\mathrm{e}{-15}$
after next chase	$1.6920\,\mathrm{e}{-14}$	$5.3346\,\mathrm{e}{-15}$
before fusion	$1.2456\,\mathrm{e}{-14}$	$3.0242\,\mathrm{e}{-14}$

TABLE 4.1
Perturbation of the shift during misfit chasing and misfit exchange, for different choices of f_{1n} and $r_{100,100}$.

n	25	50	100	200
no. iterations / n	4.8626 e+00	4.6840 e+00	4.5745 e+00	4.5687 e+00

TABLE 4.2
Average number of iterations per eigenvalue for a random Hamiltonian matrix of size $2n$.

4.3. The effect of different position vectors. Different position vectors can influence the convergence behavior of the extended QR algorithm [30]. We test this fact here for the extended K-Hamiltonian QR algorithm.

We generate a K-Hamiltonian Hessenberg matrix with $\operatorname{rank} F = 1$ and prescribed eigenvalues $[-2 : 1/n : -1 - 1/n, 1 + 1/n : 1/n : 2]$. To do so, we use an inverse eigenvalue problem based on rotation chasing described in [25] that we adapted to the K-Hamiltonian setting. We arrange the eigenvalues on the diagonal obeying the K-Hamiltonian structure, then apply a real rotation to the rows n and $n+1$ to bring F to rank 1. Finally we apply the inverse eigenvalue code based on [25]. The result is a K-Hamiltonian Hessenberg matrix. Now we apply a random unitary K-symplectic matrix, $\left[\begin{smallmatrix} Q & 0 \\ 0 & \Phi Q \Phi \end{smallmatrix}\right]$, to H and reduce the resulting matrix back to extended K-Hamiltonian Hessenberg form with the algorithm from [15]. Thus, we generate extended K-Hamiltonian Hessenberg matrices with eigenvalues close to a desired distribution. Unfortunately the inverse eigenvalue problem perturbs the eigenvalues by about 10^{-12}. Hence, we use the absolute backward error of the Schur decomposition as accuracy measure and not a forward error as the distance to the given eigenvalues.

In Figure 4.1 the absolute backward error (solid line) and the number of iterations divided by n (dashed line) are plotted. We observe that the shape does not influence the absolute backward error of the Schur decomposition. However, the Hessenberg

and the inverse Hessenberg shape require significant less iterations than the CMV and the random shape. The inverse Hessenberg shape is slightly below 3 iterations per eigenvalue.

Next, we tested the extended Hamiltonian QR algorithm for the eigenvalue distributions $[-1 : 1/n : -1/n, 1/n : 1/n : 1]$, see Figure 4.2, and $1./[-1 : 1/n : -1/n, 1/n : 1/n : 1]$, see Figure 4.3: the results are very similar. We also tested random eigenvalues based on a uniform distribution showing the same behavior as the other experiments.

We find it surprising that the inverse Hessenberg shape always requires the least number of iterations. This is not in accordance with the observation for extended QR algorithms on general matrices: in [30], the inverse Hessenberg pattern performed worse than the standard Hessenberg pattern when the eigenvalues were distributed as in Figure 4.3. A better understanding of how a particular chosen shape influences the convergences behavior in the K-Hamiltonian setting will be subject of future research.

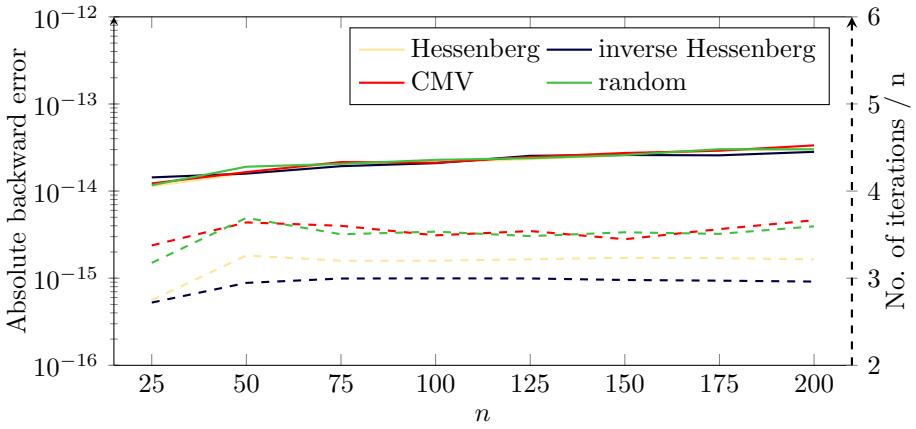


FIG. 4.1. *Absolute backward error (left scale, solid line) and number of iterations (right scale, dashed line) for different shapes for equally spaced eigenvalues (*$[-2 : 1/n : -1 - 1/n, 1 + 1/n : 1/n : 2]$*).*

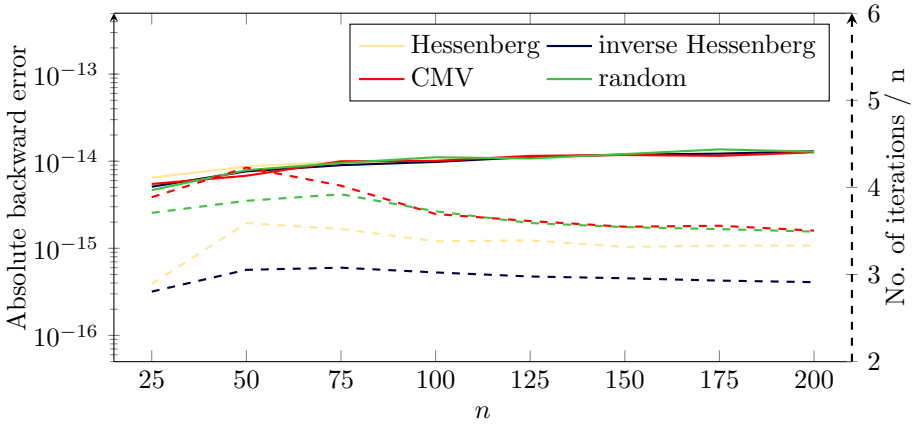


FIG. 4.2. *Absolute backward error (left scale, solid line) and number of iterations (right scale, dashed line) for different shapes for equally spaced eigenvalues (*$[-1 : 1/n : -1/n, 1/n : 1/n : 1]$*).*

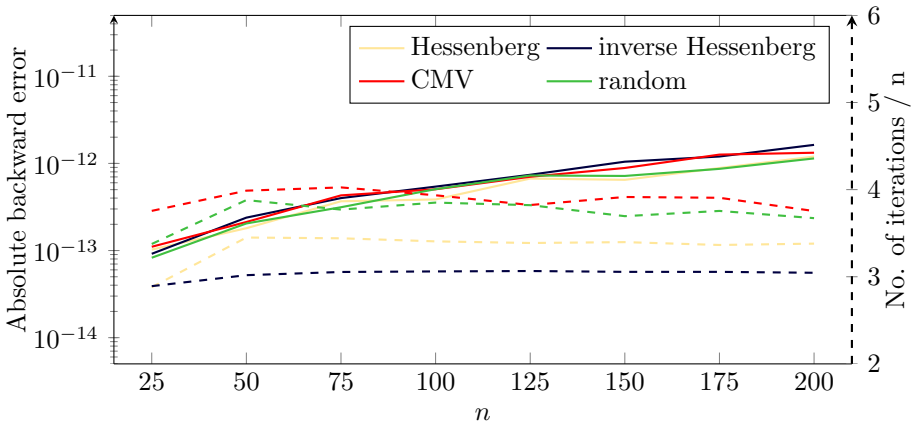


FIG. 4.3. *Absolute backward error (left scale, solid line) and number of iterations (right scale, dashed line) for different shapes for equally spaced eigenvalues* $(1./[-1:1/n:-1/n, 1/n:1/n:1])$.

Example	n	shape	ρ_{red}	# it./ev.	ρ_{QR}
14	4	Hessenberg	9.25 e−16	8.25	5.44 e−15
14	4	inv. Hess.	4.36 e−16	5.50	6.23 e−15
14	4	CMV	4.25 e−16	5.50	9.19 e−15
14	4	random	7.70 e−16	5.50	5.49 e−15
18	100	Hessenberg	1.00 e−14	3.09	1.01 e−14
18	100	inv. Hess.	1.32 e−14	2.84	5.75 e−15
18	100	CMV	6.55 e−14	3.24	9.18 e−15
18	100	random	2.44 e−13	3.30	1.16 e−14

TABLE 4.3

Number of iterations per eigenvalue (# it./ev.), relative backward error for the reduction to extended Hamiltonian Hessenberg form (ρ_{red}) and for the computation of the Hamiltonian Schur form (ρ_{QR}) for CAREX examples 14 and 18 and different patterns.

4.4. CAREX. We test the reduction to extended Hamiltonian Hessenberg form and the computation of the Hamiltonian Schur form for two examples, Example 14 and Example 18, from the benchmark collection CAREX for continuous algebraic Riccati equations [6]. The other examples are either very small ($n = 2$), have rank $F > 1$, or are already almost in Hamiltonian Schur form. The examples from CAREX are parameter dependent, hence we used the standard parameter setting.

The results are shown in Table 4.3, there ρ_{red} is the relative backward error for the reduction to extended Hamiltonian Hessenberg form and ρ_{QR} the relative backward error of the Hamiltonian Schur form computed by the extended Hamiltonian QR algorithm. If we compare the different shapes, then we see the same picture as in the tests above: the inverse Hessenberg shape needs the least iterations followed by the Hessenberg shape for example 18. Example 14 is arguably too small to draw conclusions from the iterations per eigenvalue.

5. Conclusions and future work. We have presented a new structured algorithm for computing the Hamiltonian Schur form of an extended Hamiltonian Hes-

senberg matrix.

The numerical experiments are based on a simple single shift implementation. We are convinced that including well-known features, such as, aggressive early deflation [9,26], multishift or multibulge steps with blocking [10,20,31], and changing to Fortran would significantly improve the speed of the algorithm. This is the subject of further investigations.

For real Hamiltonian matrices it is also relevant to perform the QR iterations in real arithmetic to preserve the eigenvalue symmetry with respect to the real axis. This requires a double shift version of the algorithm. While chasing bulges related to more than one eigenvalue in an extended QR algorithm is easily possible [31], the bulge exchange is much more complicated and is therefore subjected to future research.

REFERENCES

[1] G. S. Ammar and V. Mehrmann, *On Hamiltonian and symplectic Hessenberg forms*, Linear Algebra and its Applications, 149 (1991), pp. 55–72.

[2] J. L. Aurentz, T. Mach, R. Vandebril, and D. S. Watkins, *eiscor – eigenvalue solvers based on core transformations*. https://github.com/eiscor/eiscor, 2014–2015.

[3] P. Benner, *Symplectic balancing of Hamiltonian matrices*, SIAM Journal on Scientific Computing, 22 (2001), pp. 1885–1904.

[4] P. Benner and D. Kressner, *Balancing sparse Hamiltonian eigenproblems*, Linear Algebra and its Applications, 415 (2006), pp. 3–19.

[5] P. Benner, D. Kressner, and V. Mehrmann, *Skew-Hamiltonian and Hamiltonian eigenvalue problems: Theory, algorithms and applications*, in Proceedings of the Conference on Applied Mathematics and Scientific Computing, Z. Drmač, M. Marušić, and Z. Tutek, eds., Springer Netherlands, 2005, pp. 3–39.

[6] P. Benner, A. J. Laub, and V. Mehrmann, *A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: Continuous-time case*, Preprints on Scientific Parallel Computing SPC 95–22, TU Chemnitz, 1995.

[7] P. Benner, V. Mehrmann, and H. Xu, *A new method for computing the stable invariant subspace of a real hamiltonian matrix*, Journal of Computational and Applied Mathematics, 86 (1997), pp. 17–43.

[8] D. A. Bini, B. Iannazzo, and B. Meini, *Numerical Solution of Algebraic Riccati Equations*, SIAM, Philadelphia, 2012.

[9] K. Braman, R. Byers, and R. Mathias, *The multishift QR algorithm. Part I: Maintaining well-focused shifts and level 3 performance*, SIAM Journal on Matrix Analysis and Applications, 23 (2002), pp. 929–947.

[10] ———, *The multishift QR algorithm. Part II: Aggressive early deflation*, SIAM Journal on Matrix Analysis and Applications, 23 (2002), pp. 948–973.

[11] A. Bunse-Gerstner, *An analysis of the HR algorithm for computing the eigenvalues of a matrix*, Linear Algebra and its Applications, 35 (1981), pp. 155–173.

[12] R. Byers, *A Hamiltonian QR-algorithm*, SIAM Journal on Scientific and Statistical Computation, 7 (1986), pp. 212–229.

[13] M. J. Cantero, L. Moral, and L. Velazquez, *Five-diagonal matrices and zeros of orthogonal polynomials on the unit circle*, Linear Algebra and its Applications, 362 (2003), pp. 29–56.

[14] D. Chu, X. Liu, and V. Mehrmann, *A numerical method for computing the hamiltonian schur form*, Numerische Mathematik, 105 (2007), pp. 375–412.

[15] M. Ferranti, B. Iannazzo, T. Mach, and R. Vandebril, *An extended Hessenberg form for Hamiltonian matrices*, Calcolo, (2015). Accepted, 22 pages.

[16] M. Ferranti, T. Mach, and R. Vandebril, *Extended Hamiltonian Hessenberg matrices arise in projection based model order reduction*, in Proceedings in Applied Mathematics and Mechanics, vol. 15, 2015, pp. 583–584.

[17] J. G. F. Francis, *The QR Transformation a unitary analogue to the LR transformation–Part 1*, The Computer Journal, 4 (1961), pp. 265–271.

[18] ———, *The QR Transformation–Part 2*, The Computer Journal, 4 (1962), pp. 332–345.

[19] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, USA, 4th ed., 2013.

[20] L. Karlsson, D. Kressner, and B. Lang, *Optimally packed chains of bulges in multishift QR algorithms*, ACM Transactions on Mathematical Software, 40 (2014). Article no. 12.

[21] H. Kimura, *Generalized Schwarz form and lattice-ladder realizations of digital filters*, IEEE Transactions on Circuits and Systems, 32 (1985), pp. 1130–1139.
[22] D. Kressner, *Numerical methods for general and structured eigenvalue problems*, vol. 46 of Lecture Notes in Computational Science and Engineering, Springer, 2005.
[23] A. J. Laub, *A Schur method for solving algebraic Riccati equations*, IEEE Transactions on Automatic Control, 24 (1979), pp. 913–921.
[24] T. Mach, M. S. Pranić, and R. Vandebril, *Computing approximate extended Krylov subspaces without explicit inversion*, Electronic Transactions on Numerical Analysis, 40 (2013), pp. 414–435.
[25] T. Mach, M. Van Barel, and R. Vandebril, *Inverse eigenvalue problems linked to rational Arnoldi, and rational (non)symmetric Lanczos*, Journal of Computational and Applied Mathematics, 272 (2014), pp. 377–398.
[26] T. Mach and R. Vandebril, *On deflations in extended QR algorithms*, SIAM Journal on Matrix Analysis and Applications, 35 (2014), pp. 559–579.
[27] V. Mehrmann, *The Autonomous Linear Quadratic Control Problem: Theory and Numerical Solution*, vol. 163 of Lecture Notes in Control and Information Sciences, Springer, 1991.
[28] V. Mehrmann, C. Schröder, and D. S. Watkins, *A new block method for computing the hamiltonian schur form*, Linear Algebra and its Applications, 431 (2009), pp. 350–368.
[29] F. Tisseur, *Stability of structured Hamiltonian eigensolvers*, SIAM Journal on Matrix Analysis and Applications, 23 (2001), pp. 103–125.
[30] R. Vandebril, *Chasing bulges or rotations? A metamorphosis of the QR-algorithm*, SIAM Journal on Matrix Analysis and Applications, 32 (2011), pp. 217–247.
[31] R. Vandebril and D. S. Watkins, *A generalization of the multishift QR algorithm*, SIAM Journal on Matrix Analysis and Applications, 33 (2012), pp. 759–779.
[32] D. S. Watkins, *Some perspectives on the eigenvalue problem*, SIAM Review, 35 (1993), pp. 430–471.
[33] ———, *The transmission of shifts and shift blurring in the QR algorithm*, Linear Algebra and its Applications, 241–243 (1996), pp. 877–896.
[34] ———, *Bulge exchanges in algorithms of QR-type*, SIAM Journal on Matrix Analysis and Applications, 19 (1998), pp. 1074–1096.
[35] D. S. Watkins, *A case where balancing is harmful*, Electron. Trans. Numer. Anal., 23 (2006), pp. 1–4 (electronic).
[36] D. S. Watkins, *On the reduction of a hamiltonian matrix to hamiltonian schur form*, Electronic Transactions on Numerical Analysis, 23 (2006), pp. 141–157.
[37] D. S. Watkins, *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*, SIAM, Philadelphia, USA, 2007.
[38] ———, *Francis's algorithm*, American Mathematical Monthly, 118 (2011), pp. 387–403.
[39] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Numerical Mathematics and Scientific Computation, Oxford University Press, New York, USA, 1988.